Assignment 1: Recommender Systems for Rating and Purchase Prediction

Anmol Popli

PID: A53279430
apopli@ucsd.edu

Abstract. The purchase prediction task involves predicting for a given (user,item) pair whether the user purchased the item (i.e., whether it was one of the products they reviewed). I have implemented a Jaccard Similarity based approach to classify new (user,item) pairs as purchases or non-purchases, wherein the similarity measure is based on the item categories. This approach resulted in an accuracy of 65.702 % on validation set and 69.2 % on the public leaderboard.
In rating prediction, the task is to predict for a given (user,item) pair the star rating that the user will assign to the item. For this task, I trained linear as well as latent factor models. After comparing the performances of the two, I used their ensemble for my final predictions. The MSE on validation set is 1.13548 and MSE on public leaderboard is 1.13844.

1 Purchase Prediction

1.1 Methods

For a given (user,item) pair, I compare the similarity between the given item and all items purchased by the given user. Also, I compare the similarity between the given user and all users that have purchased the given item. The similarity measure used here is Jaccard similarity. Conventionally, jaccard similarity is computed between two items based on the users who purchased them, and between two users based on the items they purchased. Since the dataset is pretty sparse, this approach resulted in very low accuracies on the validation set.

Unlike this conventional use of Jaccard similarity in prediction tasks, I have computed Jaccard similarity between items based on the categories they belong to. Similarly, for users I have computed the similarity based on the categories of items they have purchased. Here, the categories I have used are 'Clothing, Shoes & Jewelry', 'Women', 'Clothing' etc, i.e., the actual category names and not the sublists of categories.

To begin with, corresponding to each item I created a list of categories to which that item belongs. Let the list for an item I be denoted by C_I. Similarly, I populated lists for users based on the categories of the items they have purchased. Let the list for a user U be denoted by C_U. Corresponding to each item I, I also prepared a list of users U_I who have purchased that item. And for each user U, I prepared a list I_U denoting items purchased by that user.

Similarity between items Let the incoming pair be denoted by (U,I). The list of items purchased by U is $I_U = \{I_U^1, I_U^2, ...\}$. I compute the jaccard similarity of the given item with each of these items.

$$J_{1,i} = Jaccard(C_I, C_{I_U^1})$$
$$J_{2,i} = Jaccard(C_I, C_{I_U^2})$$
$$...$$

and so on, where $C_{I_U^1}$ denotes the list of categories corresponding to item I_U^1. Then I take mean of all the similarity values, $J_i = mean(J_{1,i}, J_{2,i}, ...)$. In order to classify the incoming pair as purchase/non-purchase, I compare this mean value J_i to a threshold $thresh_i$. If $J_i > thresh_i$, (U,I) is classified as purchase, otherwise non-purchase. The threshold value $thresh_i$ was obtained by calculating validation accuracy while performing an iterative search over values from 0 to 1 in intervals of 0.01. The optimal value thus obtained was $thresh_i = 0.26$.

Similarity between users A similar approach is followed for similarity between users. Let the incoming pair be denoted by (U,I). The list of users who purchased I is $U_I = \{U_I^1, U_I^2, ...\}$. I compute the jaccard similarity of the given user with each of these users.

$$J_{1,u} = Jaccard(C_U, C_{U_I^1})$$
$$J_{2,u} = Jaccard(C_U, C_{U_I^2})$$
$$...$$

and so on, where $C_{U_I^1}$ denotes the list of categories corresponding to user U_I^1. As was the case with item-item similarity, I take mean of all the similarity values, $J_u = mean(J_{1,u}, J_{2,u}, ...)$. In order to classify the incoming pair as purchase/non-purchase, I compare this mean value J_u to a threshold $thresh_u$. If $J_u > thresh_u$, (U,I) is classified as purchase, otherwise non-purchase. The threshold value $thresh_u$ was obtained by calculating validation accuracy while performing an iterative search over values from 0 to 1 in intervals of 0.01. The optimal value thus obtained was $thresh_u = 0.22$.

Ensemble Performing an OR on the outputs (purchase/non-purchase: 1/0) from the item-item and user-user similarity methods resulted in an even higher validation set accuracy. This shows that satisfying any one of these similarity criteria is sufficient to predict that the purchase was made. Therefore, in order to make my final prediction of purchase/non-purchase, I perform an OR on the outputs of the two aforementioned methods. That is, if any of the above two methods predict 'purchase', the final prediction is 'purchase'. If both predict 'non-purchase', the final prediction is 'non-purchase'.

1.2 Results

Using the item-item similarity method, an accuracy of 65.505 % was achieved on the validation set. Using the user-user similarity method, an accuracy of 64.408 % was achieved on the validation set. The ensemble resulted in an accuracy of 65.702 % on validation set and 69.2 % on the public leaderboard.

2 Rating Prediction

2.1 Methods

For the rating task, I trained both a linear model $(\alpha, \beta_u, \beta_i)$ as well as a latent factor model $(\alpha, \beta_u, \beta_i, \gamma_u, \gamma_i)$. Though the linear model performs better on unseen users and items, the latent factor model is able to reduce the MSE on seen users and items when used in conjunction with linear model.

Linear Model The linear model was trained using alternating least squares optimization. I implemented early stopping to find the optimal number of epochs to train, which was found to be 1. The validation MSE was the minimum after first epoch and continuously increased in subsequent epochs. Henceforth I trained this model for only one epoch for all subsequent experiments. I tuned the regularization constant through cross-validation and found a value of $\lambda = 6.4$ to be the optimal one.

Latent Factor Model The latent factor model was trained from scratch using Stochastic Gradient Descent. For a learning rate of 0.005, I found the optimal number of epochs to train to be 35 through early stopping on validation set. Then through cross-validation, I tuned the regularization constant and the number of latent dimensions (dimensionality of γ_u and γ_i), and arrived on optimal values for regularization constant $\lambda = 1$ and number of latent dimensions $k = 55$.

2.2 Ensembles and Results

The linear model resulted in an MSE of 1.1356427 on the validation set. The latent factor model resulted in an MSE of 1.1374396 on the validation set. A possible reason for the latent factor model's higher MSE is that in case of unseen users or items, it predicts the rating using only its α and β parameters, which haven't been trained to predict on their own (i.e., in the absence of γ parameters). To deal with this issue I implemented an ensemble wherein for unseen users/items, I predict the rating using linear model with its trained α, β parameters and for seen users/items, I predict the rating using latent factor model with its trained α, β, γ parameters. Pseudo code of this ensemble is as follows:

```
if user U is unseen OR item I is unseen then
    prediction = linear_model(U,I)
else
    prediction = latent_factor_model(U,I)
```

In this ensemble, the α and β parameters of the linear model are distinct from those of latent factor model, as the two have been trained separately. This ensemble resulted in an MSE of 1.1371698 on validation set, which is an improvement over the latent factor model but still inferior to the standalone linear model. In order to further reduce my validation MSE, I implemented a weighted ensemble as follows:

```
if user U is unseen OR item I is unseen then
    prediction = linear_model(U,I)
else
    prediction = W× latent_factor_model(U,I) +(1 − W)× linear_model(U,I)
```

I tuned the weight factor W for this ensemble using cross-validation and arrived at an optimal $W = 0.4$. This weighted ensemble resulted in an MSE of 1.1354794 on the validation set, which is lower than that of standalone linear model. I use this weighted ensemble of linear and latent factor models as my final rating prediction model. Its MSE on the public leaderboard is 1.13844.